Clicker Question 1

» 2000 elements are inserted one at a time into an initially empty binary search tree using the traditional, naive algorithm. What is the maximum possible height of the resulting tree?

A. 1
B. 11
C. 21
D. 1999
E. 2000

Red Black Trees

A BST with more complex algorithms to ensure balance

» Each node is labeled as Red or Black.

» Path: A unique series of links (edges) traverses from the root to each node.
  – The number of edges (links) that must be followed is the path length

» In Red Black trees paths from the root to elements with 0 or 1 child are of particular interest

Binary Search Trees

» Average case and worst case Big O for
  – insertion
  – deletion
  – access

» Balance is important. Unbalanced trees give worse than log N times for the basic tree operations

» Can balance be guaranteed?
Paths to Single or Zero Child Nodes

- How many?

![Red Black Tree Diagram](image)

Red Black Tree Rules

1. Every node is colored either red or black
2. The root is black
3. If a node is red its children must be black. (a.k.a. the red rule)
4. Every path from a node to a null link must contain the same number of black nodes (a.k.a. the path rule)

Example of a Red Black Tree

- The root of a Red Black tree is black
- Every other node in the tree follows these rules:
  - Rule 3: If a node is Red, all of its children are Black
  - Rule 4: The number of Black nodes must be the same in all paths from the root node to null nodes

![Red Black Tree Example](image)

Red Black Tree?

![Red Black Tree Question](image)
Clicker Question 2

- Is the tree on the previous slide a binary search tree? Is it a red black tree?
  - BST?  Red-Black?
  A. No  No
  B. No  Yes
  C. Yes  No
  D. Yes  Yes

Clicker Question 3

- Is the tree on the previous slide a binary search tree? Is it a red black tree?
  - BST?  Red-Black?
  A. No  No
  B. No  Yes
  C. Yes  No
  D. Yes  Yes

Implications of the Rules

- If a **Red** node has any children, it must have two children and they must be Black. (Why?)
- If a **Black** node has only one child that child must be a **Red** leaf. (Why?)
- Due to the rules there are limits on how unbalanced a **Red** Black tree may become.
  - on the previous example may we hang a new node off of the leaf node that contains 0?
Properties of Red Black Trees

- If a Red Black Tree is complete, with all Black nodes except for Red leaves at the lowest level the height will be minimal, \( \sim \log N \)
- To get the max height for \( N \) elements there should be as many Red nodes as possible down one path and all other nodes are Black
  - This means the max height would be approximately \( 2 \times \log N \)
  - see example on next slide

Maintaining the Red Black Properties in a Tree

- Insertions
- Must maintain rules of Red Black Tree.
- New Node always a leaf
  - can't be black or we will violate rule 4
  - therefore the new leaf must be red
  - If parent is black, done (trivial case)
  - if parent red, things get interesting because a red leaf with a red parent violates rule 3

Insertions with Red Parent - Child

Must modify tree when insertion would result in Red Parent - Child pair using color changes and rotations.
Case 1

- Suppose sibling of parent is Black.
  - by convention null nodes are black
- In the previous tree, true if we are inserting a 3 or an 8.
  - What about inserting a 99? Same case?
- Let X be the new leaf Node, P be its Red Parent, S the Black sibling and G, P's and S's parent and X's grandparent
  - What color is G?

### Case 1 - The Picture

Relative to G, X could be an inside or outside node. Outside -> left left or right right moves Inside -> left right or right left moves

Fixing the Problem

If X is an outside node a single rotation between P and G fixes the problem.
A rotation is an exchange of roles between a parent and child node. So P becomes G's parent. Also must recolor P and G.

Single Rotation

Apparent rule violation?
Case 2

- What if $X$ is an inside node relative to $G$?
  - a single rotation will not work
- Must perform a double rotation
  - rotate $X$ and $P$
  - rotate $X$ and $G$

After Double Rotation

Apparent rule violation?

Case 3

Sibling is Red, not Black

- Must perform single rotation between parent, $P$ and grandparent, $G$, and then make appropriate color changes

Fixing Tree when $S$ is Red

Any problems?
More on Insert

- Problem: What if on the previous example G's parent had been red?
- Easier to never let Case 3 ever occur!
- On the way down the tree, if we see a node X that has 2 Red children, we make X Red and its two children black.
  - if recolor the root, recolor it to black
  - the number of black nodes on paths below X remains unchanged
  - If X's parent was Red then we have introduced 2 consecutive Red nodes. (violation of rule)
  - to fix, apply rotations to the tree, same as inserting node

Example of Inserting Sorted Numbers

- 1 2 3 4 5 6 7 8 9 10

Insert 1. A leaf so red. Realize it is root so recolor to black.

Insert 2

make 2 red. Parent is black so done.

Insert 3

Insert 3. Parent is red. Parent's sibling is black (null) 3 is outside relative to grandparent. Rotate parent and grandparent
Insert 4
On way down see 2 with 2 red children. Recolor 2 red and children black. Realize 2 is root so color back to black

When adding 4 parent is black so done.

Finish insert of 5

Insert 5
5's parent is red. Parent's sibling is black (null). 5 is outside relative to grandparent (3) so rotate parent and grandparent then recolor

Insert 6
On way down see 4 with 2 red children. Make 4 red and children black. 4's parent is black so no problem.
Finishing insert of 6

6's parent is black so done.

Insert 7

7's parent is red. Parent's sibling is black (null). 7 is outside relative to grandparent (5) so rotate parent and grandparent then recolor

Finish insert of 7

Insert 8

On way down see 6 with 2 red children. Make 6 red and children black. This creates a problem because 6's parent, 4, is also red. Must perform rotation.
Still Inserting 8

Recolored now need to rotate

Finish inserting 8

Recolored now need to rotate

Insert 9

On way down see 4 has two red children so recolor 4 red and children black. Realize 4 is the root so recolor black

Finish Inserting 9

After rotations and recoloring
Insert 10

On way down see 8 has two red children so change 8 to red and children black.

Insert 11

Again a rotation is needed.

Finish inserting 11